Internet Application Traffic Classification Using Fixed IP-Port

Sung-Ho Yoon, Jin-Wan Park, Jun-Sang Park, Young-Seok Oh, and Myung-Sup Kim

Dept. Of Computer and Information Science, Korea University, Korea
{sungho_yoon,jinwan_park,junsang_park,youngsuk_oh, tmskim}@korea.ac.kr

Abstract. As network traffic is dramatically increasing due to the popularization of Internet, the need for application traffic classification becomes important for the effective use of network resources. In this paper, we present an application traffic classification method based on fixed IP-port information. A fixed IP-port is a {IP, protocol, port} triple dedicated to only one application, which is automatically collected from the behavior analysis of individual applications. We can classify the Internet traffic accurately and quickly by simple packet header matching to the collected fixed IP-port information. Therefore, we can construct a lightweight, fast, and accurate real-time traffic classification system than other classification method. In this paper we propose a novel algorithm to extract the fixed IP-port information and the system architecture. Also we prove the feasibility and applicability of our proposed method by an acceptable experimental result.

Keywords: Traffic monitoring and analysis, Traffic classification, Application identification, Fixed IP-port.

1 Introduction

Nowadays, network traffic is rapidly increasing because of the increase of Internet users and high speed network links. It is caused by not only traditional Internet service such as WWW, FTP, and e-mail, but also development of various multimedia services: for example, multimedia streaming, P2P(peer-to-peer) file sharing, gaming, and so on. Consequently, the total Internet traffic volume has rapidly grown, which makes the need of traffic monitoring and classification inevitable for efficient network management [1, 2].

Network traffic classification is a series of processes to capture packets from a target network link and determine the identity of packets from the perspective of categorization. For example, it is to figure out the corresponding application names of the captured network packets. The result of classification can be effectively utilized on network management and control. We can accurately block or shrink the traffic from applications individually. Also, it can improve the understanding of network user's activity. Traffic classification can be carried out under various categorization perspectives such as application-layer protocol, traffic types, application programs, and so on.

C.S. Hong et al. (Eds.): APNOMS 2009, LNCS 5787, pp. 21–30, 2009.

Previously, many methodologies have been proposed for accurate traffic classification such as well-known port-based, signature-based and machine learning-based, and so on. All above methods, however, have some limitations in applying them to classifying the traffic on a real operational network. Well-known port-based method using IANA port information cannot classify some applications which use dynamic or unknown ports [3]. Signature-based [4] method requires formerly created payload signatures and cannot be applied to the encrypted payloads. Machine learning-based method [5] also has limitation such as requisite of prior learning and difficult of adaptation to newly emerging application. The weakest point of last two methods is much processing overhead in applying to real-time classification system.

This paper proposes an application traffic classification method based on fixed IP-port information. A fixed IP-port is a {IP, protocol, port} triple dedicated to only one application, which is automatically collected from the behavior analysis of individual applications. A fixed IP-port for an application is absolutely accurate because it is extracted from end hosts where the application process is actually running to create traffic. It can overcome the limitations of the currently existing methods. We can classify the Internet traffic effectively and quickly by simple packet header matching to the fixed IP-port information. Therefore, we can construct a lightweight, fast, and accurate real-time traffic classification system than other classification method. In this paper we propose a novel algorithm to extract the fixed IP-port information and the system architecture. Also we prove the feasibility and applicability of our proposed method by an acceptable experimental result.

The rest of the paper is organized as follows. Section 2 describes the definition of the fixed IP-port. The methodology to automatically extract the fixed IP-port for each application is described in section 3. Section 4 describes the traffic classification system based on the fixed IP-port. In section 4, an experiment and result are presented. Section 5 summaries this paper and shows possible future work.

2 The Fixed IP-Port, What Is It?

A fixed IP-port is defined as a {IP address, port number, transport protocol} triple which is dedicated to a single application only. Actually it is server socket information which is open to the public for a service delivery. For example, in case an application requires a user authentication before providing service, the login server's {IP, protocol, port} triple is to be one of the fixed IP-ports of that application.

2.1 Motivation of the Fixed IP-Port

The port number based traffic classification is no longer accepted on current highly dynamic network environment. However we believe that {IP, protocol, port} based method might be acceptable, because most of the current popular Internet applications have at least one dedicated server and port for the application. The best example of this is the login server and port for user authentication. In some circumstances, we have to divide the traffic with the same application protocol into each specific service or application. In this case the {IP, protocol, port} information can be utilized effectively, because each service is provided from different servers with different IPs even though they use same protocol and port number.

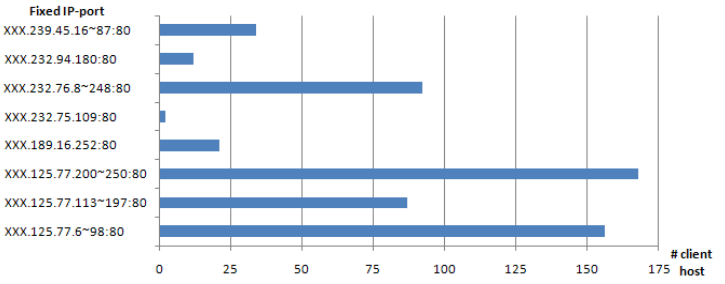

Fig. 1. The Fixed IP-ports of the gom.exe from Gretech Inc

To convince ourselves we made a simple investigation on our campus network. We selected one application called Gom (gom.exe) [7] from Gretech Inc. The Gom is a multimedia player which is popular in Korea and Japan. We collected all TCP and UDP connections from the Gom during 7 days from our campus network using KU-MON system and analyzed the remote IPs and ports. The KU-MON is a real-time traffic monitoring system developed by our research group.

Fig. 1. shows the number of local hosts having a connection to the remote {IP, port} pairs when the host plays the Gom application. All of the remote {IP, port} pairs listed in Fig.1 are not used by other applications. The {IP, port} pairs in Fig. 1 are used only Gom applications. The result shows that there are some {IP, port} pairs used by Gom application only. Some of them appear every time the Gom application is running while other appears irregularly. This result makes us confirm the existence of fixed IP-port. BLINC [3] explains these {IP, port} pairs as “farm”.

2.2 The Fixed IP-Port Based Traffic Classification

If we have a collection of fixed IP-port triple for every application, by simple packet header matching we can easily identify the application name of every traffic data. But it is impossible because of the dynamically assigned port numbers and P2P applications.

However, if we can find at least one fixed IP-port triple for each application, we can determine that an application is running on the host or not. We know all the applications running on a host by this simple matching. For the non-determined traffic flows we can easily develop a method to determine the application name, because that should be one of the applications on the host and we know all of the applications running on the host.

How to collect the fixed IP-port triple for every application? In this paper we propose a novel method to automatically extract the fixed IP-port information by the flow behavior analysis of individual applications.

3 Fixed IP-Port Extraction Algorithm

In this chapter we describe the proposed fixed IP-port extraction algorithm. The Input to this algorithm is a set of flows determined the target applications, and the output is a set of fixed IP-port triples. The output could be nothing if the target application does not have one. The proposed algorithm is applied to each application separately.

The key of the proposed algorithm is to select the fixed IP-port triples among all the IP-port triples appear in the flow records of a target application. To achieve this goal, we distinguished IP-port triples of an application into three types: permanent IP-port, temporal IP-port, and dynamic IP-port. The permanent IP-port is an IP-port triple which is permanently used for the target application only. The best example of this is the login server's IP-port triple. The dynamic IP-port is an IP-port triple which is dynamically selected one and never shared by other flows. An example of this type is the client's IP-port triple in HTTP connection. The temporal IP-port is an IP-port which is a dynamic one, but is shared by other flows for certain time interval. An example of this type is the temporal P2P host IP-port, which is used only small amount of time for the P2P application and used by other application later.

The dynamically selected IP-ports should be eliminated while the permanently selected IP-ports should be defiantly selected as the fixed IP-port. The temporally selected IP-ports should be effectively un-selected, which is not the fixed IP-port in our definition.

3.1 Considerations

The fixed IP-port Extraction methodology proposed in this paper should consider the follows three performance metrics.

Coverage: We have to find the fixed IP-port triples as many as possible. Our methodology should find at least one fixed IP-port for each application.

Accuracy: The selected fixed IP-port triples should be accurate enough which minimize the FP(false positive) and FN(false negative) [6].

Overhead: We have to minimize the memory and processing overhead of the fixed IP-port extraction system. Among huge amount of IP-port triples it is important to eliminate the dynamic IP-port triples as quickly as possible to reduce memory space and processing burden.

3.2 The Fixed IP-Port Extraction Algorithm

Fig. 2 describes the overall algorithm for the proposed fixed IP-port extraction as a state transition diagram. These are the most important in this paper.

All IP-port triples exist in one of the five states (Discard, New, Candidate, Fixed, Time_wait). The IP-port triple in the Fixed state is the fixed IP-port for the target application. As aforementioned this extraction algorithm executes separately for each target application. The input to the algorithm for an application is all the flow records whose application names are determined.

Initially all possible IP-port triples exist in the Discard state, which means there is no fixed IP-port for a target application. Actually no information is stored in the Discard state. If an IP-port appears in a flow record, the IP-port moves to the New state. If an IP-port satisfies some specific conditions, it moves to the Candidate state and the subsequent Fixed state. The IP-ports in the Fixed state are the fixed IP-port for the corresponding application, which is used for the traffic classification.

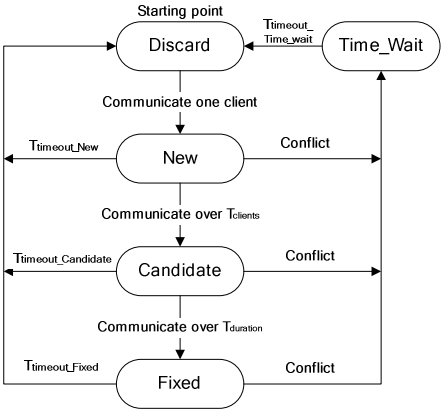

Fig. 2. The State Transition Diagram for Fixed IP-port Extraction

In the New, Candidate, and Fixed states, we prevent that an IP-port remains in the states forever using conflict and timeout check. The Time_wait state exists for a conflicted IP-port with other application to reenter the New state again just after eleminated. This causes high system burden because the reentered conflicted IP-port might be eliminated again soon by another conflict flow record.

We have to consider several conditions for the state transition. We use three threshold values for state transition conditions as follows.

$\mathbf{T}_{clienthost}$ It is number of different client hosts to a single IP-port triple. If one IP-port is fixed IP-port, all client flows using the IP-port belong to same application. Therefore, an IP-port triple having over $T_{clienthost}$ clients can be determined as the fixed IP-port. If $T_{clienthost}$ is increase, accuracy, also, increase, but, completeness decreases.

$\mathbf{T}_{duration}$ It means usage time (last usage time – registered time). If we use only $T_{clienthost}$ to determine the fixed IP-port, we can make a mistake to determine a temporal IP-port such as P2P application as fixed IP-port. However, we can distinguish these temporal IP-ports by using usage time. The feature of P2P application is short usage time. Therefore, we make $\mathbf{T}_{duration}$ and use it to extract real fixed IP-port.

$\mathbf{T}_{timelimit}$ Initially, all IP-ports wait in memory until excess $\mathbf{T}_{clienthost}$ and $\mathbf{T}_{duration}$. Unfortunately, we cannot store all IP-port triples forever due to the limit of physical memory. It, also, causes fail of real time classification system because of system overhead. Therefore, some IP-ports which over time limit without status change should be removed.

These thresholds make our extracting algorithm flexible and efficient in various environments. The main point in Fixed IP-port Algorithm is the transition among states in the state transition diagram. The proposed algorithm implements according to Fig. 3.

The code is divided in two parts. First part is to register new IP-port to the IPT as a New state or to update the condition of the existing IP-ports according to the new flow records: (conflict, number of client, and last usage time). If new flow record does

```
1:   procedure Fixed IP-port Extraction for an Application A
2:       IPT ← IP-port Table
3:
4:
5:       for each input flow record do
6:            search the corresponding IP-port record in IPT
7:           if found in IPT then
8:                update the status of the record;
9:           else add a new record with the state as New;
10:      end for
11:
12:      for each flow record in IPT do
13:           Check the state of the record
14:           if state == New then
15:                if conflict then move to Time_Wait;
16:                else if over Tclienthosts then move to Candidate;
17:                  else if over Ttimelimit then move to Discard;
18:           else if State == Candidate then
19:                  if conflict then move to Time_Wait;
20:                else if over Tduration then move to Fixed;
21:                  else if over Ttimelimit then move to Discard;
22:            else if State == Fixed then
23:                if conflict then move to Time_Wait;
24:                else if over Ttimelimit then move to Discard;
25:           else if State == Time_Wait then
36:                if over Ttimelimit then   move to Discard;
37:      end for
38:
     end procedure
```

Fig. 3. The Pseudo Code for the Fixed IP-port Extraction Algorithm

not exist in the IPT then a record is added to the IPT. If the existing record and new record are belongs to different application, the conflict value is checked. The new and existing record belong to the same application, the usage time is updated.

The Second part is to transfer the state of IP-port record in the IPT according to status changed in the previous step in the status transition diagram. If a record in any state is not accessed over the $T_{timelimte}$ time, the record moves to the Discard state. If a record is conflicted with other application then it moves to the Time_Wait state. If a record is accessed over $T_{clienthost}$ during over $T_{duration}$ time, then the record moves to the Fixed state. The IP-port triples in the Fixed state are the fixed IP-ports of the application as aforementioned.

4 The Traffic Classification System Based on the Fixed IP-Port

Fig. 4. shows the fixed IP-port based traffic classification system we developed in our campus network. This system consists of three subsystems: the fixed IP-port extraction system, the application traffic classification system, and the traffic verification system. It is a part of our KU-MON system.

The fixed IP-port extraction system is the realization of the proposed fixed IP-port extraction algorithm, which is described in chapter 3. The traffic classification system classifies Internet traffic into corresponding applications by simple header matching using the fixed IP-port records. The traffic verification system verifies the classification results in various points of view.

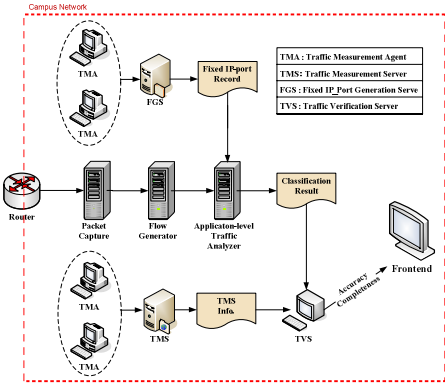

Fig. 4. The Fixed IP-port Extraction System Structure

4.1 Fixed IP-Port Extraction System

The fixed IP-port extraction system uses the TMA(Traffic Measurement Agent) [6] records and flow data as its input data. TMA is a tiny agent program which is installed in end-hosts, gathers network socket information and periodically sends it to the gathering server. Whenever a new socket is created by an application process, the TMA gathers the socket information, and sends them to the FGS and TMS. We installed TMAs at about 300 end-hosts in our campus network. TMA records from about 100 end-hosts are used by the fixed IP-port extraction system and other TMA records are used by the traffic verification system.

The fixed IP-port extraction system generates a ground truth flows from the TMA records and flow records. Based on the ground truth flows the system extract fixed IP-port according to the aforementioned algorithm.

4.2 Traffic Classification System

This system is simple. We can classify traffic with flow-based real-time classification because fixed IP-port is lightweight. Firstly, packets are captured from target network link (Packet Capture). And then, we make them into flows (Flow Generator). In the Application-level Traffic Analyzer, we classify traffic according to fixed IP-port by compare with just flow header which contains 5-tuples. Therefore, it is very lightweight and simple.

4.3 Traffic Verification System

The verification is the measurement of accuracy of classification by comparing ground truth with the result of the classification algorithm. So, the most important is to create correct ground truth. Previous papers used various ground truth data for verification of their algorithm. BLINC[3] uses result of payload-base signature classification; and Constantinou et al.[8] used result of well-known port-based classification. These ways to create ground truth data have some weakness. They used uncertain ground truth, result of other classification method, for valid their algorithm.

These are not suitable for correct verification. To make up previous uncertain way for creating ground truth, Szabó et al [9] proposed better way that installs driver in end host, creating traffics, and checks identification of all user traffic. It seems be cover the shortage of ground truth. However, it cannot actually create enough ground truth because of the need of installation on all end-hosts. It causes large overhead to end-hosts, so, they would be reluctant to this. In this paper we propose verification using TMA. For correct verification, we divide TMA group into two groups, one is for creation of Fixed IP-port, and another is for verification of our algorithm because we use TMA on both extraction and verification.

Correct ground truth is important, in a same manner, it is also important to define evaluation parameters for algorithm verification. This paper shows accurate evaluating parameters for verification of classification algorithm. Previously, [10] defines above some parameters, but we add more to make accurate and detail.

The parameters consist of three parts such as coverage, accuracy, and completeness. First of all, Coverage means the number of application identified by the algorithm. Second parameter is accuracy. It means how accurate the result comparing with ground truth. Accuracy might have various ranges of verification according to the range of ground truth data. Accuracy consists of overall accuracy and individual accuracy. Especially, individual accuracy has FP(False Positive) and FN(False Negative). The FP of application X means that the algorithm classifies that some traffic is application X, but it is false. The FN of application X means that the algorithm classifies that some traffic is not application X, but it is false. Especially, FN divides FN-Unclassified and FN-Mis_Classification. The former is that the algorithm cannot classify application X. The latter is the algorithm classifies application X to other, incorrectly. [10] states that the latter (FN-Mis_Classification) is more harmful in network management. The last one is completeness which means the rate of traffic classified by the algorithm.

The Frontend of the verification system provides the following information to network managers. First of all, it shows us about basic information of the traffic. Secondly, it also shows evaluation of the algorithm such as completeness. Last is result of verification. All three parts are reported on Web every minute. So, we can check the algorithm.

5 Experiment and Results

We applied the proposed the Fixed IP-port extraction algorithm in our campus network traffic. We extract the fixed IP-ports during 12 days, and then classified traffic during the next whole day.

We determined the threshold values during the first 6 days of our 13 day experiment. To get efficient threshold values, we checked the state remaining time and P2P server usage time. Fig. 6. represents cumulative distribution function (CDF) of NC (candidate register time – new register time), New state remain time, and usage time of IP-port using P2P application. According to CDF, to eliminate dynamic and temporal IP-ports, $\mathbf{T}_{\text{duration}}$ is set to 24 hours. To reduce the overhead of remaining records, $\mathbf{T}_{\text{timelimit}}$ is set to 48 hours. $\mathbf{T}_{\text{clienthost}}$ is set to 2 for high coverage.

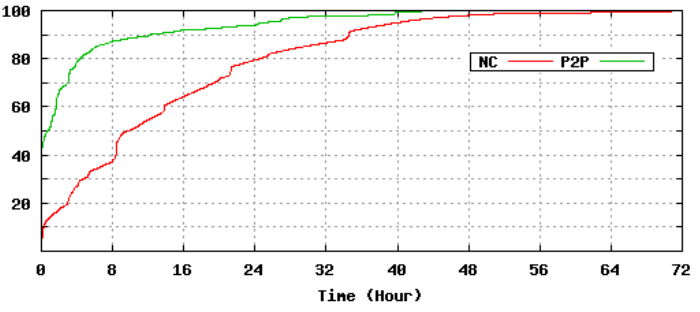

Fig. 6. CDF graphes for state usage time to get $\mathbf{T}_{\text{timelimit}}$ and $\mathbf{T}_{\text{duration}}$

Table 2. The result of traffic classification using Fixed IP-port

Completeness	28.8%(flow)
Accuracy	95.39%(flow)
Overhead	231,705(fixed)/238,543(total)
Coverage	346(fixed)/354(total)

We examined total 354 applications to extract fixed IP-ports. As a result, we extracted fixed IP-ports in 346 applications (about 98%). The total number of fixed IP-port extracted was 231,705.

Table 2. also shows the result of traffic classification using these Fixed IP-port and its accuracy. It shows the high accuracy but somewhat low completeness of the result, which means that most of applications use dynamically selected ports which should be determined using other classification method.

We extract the fixed IP-ports in most of all application. But some applications like a traditional client-server applications (ftp, ssh, mail) does not have fixed IP-port while most of modern applications does. Concerning the overhead, most IP-port remained in the Candidate state moved to the Fixed state in our system, which indicates the low overhead of our system.

6 Conclusion and Future Work

Real-time Internet traffic classification is inevitable for the efficient traffic management of network. Many of the currently existing methodologies for traffic classification have some difficulties in adopting in real operational network because of its complexity. So, this paper proposed a fixed IP-port based traffic classification approaches. A fixed IP-port is a {IP, protocol, port} triple dedicated to only one application, which is automatically collected from the behavior analysis of each applications. In this paper we proposed a novel algorithm to extract fixed IP-ports using flexible threshold values which make this algorithm adapt to various environment. Therefore, we can construct a lightweight, fast, and accurate real-time traffic classification system than other classification method.

In the future, we will extract the fixed IP-ports for more applications, and utilize it in the very first step of our real-time traffic classification system. In addition, we are interested in verification for traffic classification based on application.

Acknowledgments. This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2007-331-D00387).

References

1. Kim, M.-S., Won, Y.J., Hong, J.W.-K.: Application-Level Traffic Monitoring and an Analysis on IP Networks. ETRI Journal 27(1) (February 2005)
2. Sen, S., Wang, J.: Analyzing peer-to-peer traffic across large networks, Internet Measurement Conference (IMC). In: Proc. Of the 2nd ACM SIGCOMM Workshop on Internet measurement, pp. 137–150 (2002)
3. Karagiannis, T., Apagiannaki, K.P., Aloutsos, M.F.: BLINC: Multilevel Traffic Classification in the Dark. In: Proc. of ACM SIGCOMM (August 2005)
4. Sen, S., Spatscheck, O., Wang, D.: Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures. In: WWW 2004, New York, USA (May 2004)
5. Zander, S., Nguyen, T., Armitage, G.: Automated Traffic Classification and Application Identification using Machine Learning. In: LCN 2005, Sydney, Australia, November 15-17 (2005)
6. Park, B.-C., Won, Y.J., Kim, M.-S., Hong, J.W.: Towards Automated Application Signature Extraction for Traffic Identification. In: Proc. of the IEEE/IFIP Network Operations and Management Symposium (NOMS) 2008, Salvador, Bahia, Brazil, April 7-11, pp. 160–167 (2008)
7. GOM Player: Multimedia Player Service, `http://www.gretech.com/`
8. Constantinou, F., Mavrommatis, P.: Identifying known and unknown peer-to-peer traffic. In: IEEE International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, July 2006, pp. 93–102 (2006)
9. Szabó, G., Orincsay, D., Malomsoky, S., Szabó, I.: On the validation of traffic classification algorithms. In: Claypool, M., Uhlig, S. (eds.) PAM 2008. LNCS, vol. 4979, pp. 72–81. Springer, Heidelberg (2008)
10. Risso, F., Baldi, M., Morandi, O., Baldini, A., Monclus, P.: Lightweight, Payload-Based Traffic Classification: An Experimental Evaluation. In: Proceeding of Communications, ICC 2008, IEEE International Conference (2008)